# 阿里云ApsaraDB诊断报告

# 1 文档简介

## 1.1 文档目的

本文档描述了阿里云客户支持服务对当前问题状态的理解和认识，并给出相关建议和结论。

本文档既用于让客户对当前问题有全面的了解，也用于协调各相关人员对问题状态达成统一的认识。

## 1.2 问题类别定义

| 编号 | 类别 | 说明 |
| --- | --- | --- |
| 1. | 应用问题 | 应用问题包含开发、设计等问题。 |
| 2. | 操作使用问题 | 操作使用问题包含操作不规范,操作失误等问题。 |
| 3. | 产品问题 | 产品问题包含软件产品、硬件产品及第三方产品问题。 |
| 4. | 未知问题 | 未知问题指当前无法判断的问题类别。 |

## 1.3 问题严重级别定义

以下为我们对系统环境中问题的严重级别定义：

| 编号 | 类别 | 说明 |
| --- | --- | --- |
| S1 | 系统失败，任何处理都不能继续。 | |
| S2 | 不能够继续选定的任务处理。 | |
| S3 | 受限制的功能能力，但处理可以继续。 | |
| S4 | 目前可以忽略的小问题。 | |

# 2 规格配置

| 配置项 | 规格名称 |
| --- | --- |
| 实例名 | rr-2ze83o0z0lc4tq346 |
| 数据库类型 | mysql |
| 数据库版本 | 5.6 |
| 规格名称 | MySQL新规格8核16G |
| 存储空间(M) | 204800 |
| 最大连接数 | 4000 |
| 最大IOPS | 8000 |
| QPS参考值 | 999999 |

**综合评分：***74*分

# 3 系统状态

**时间范围** *2019-01-15 03:00* 至 *2019-01-15 18:01*


| 资源名称 | 资源状态 | 最大值 | 最小值 | 平均值 | 参考值 |
|---|---|---|---|---|---|
| CPU利用率 | 空闲 | 10.70% | 0.00% | 2.36% | 40%~60% |
| 内存利用率 | 空闲 | 55.10% | 55.10% | 55.10% | 60%~80% |
| 磁盘IOPS | 空闲 | 8.10% | 0.10% | 0.56% | 40%~80% |
| 磁盘空间 | 空闲 | 52.20% | 52.00% | 52.07% | 60%~80% |
| 连接数 | 空闲 | 1.60% | 0.50% | 0.95% | 40%~80% |

# 4 问题与建议

## 4.1 死锁

*没有发现死锁。*

## 4.2 锁等待

*没有发现锁等待。*

## 4.3 实例安全

*没有发现相关问题。*

## 4.4 连接数

*没有发现相关问题。*

## 4.5 内存

*没有发现相关问题。*

## 4.6 实例空间

### 4.6.1 其它空间

| 严重级别 | S4 |
| --- | --- |
| 问题 | 其它数据空间超过5G |
| 建议 | 1. 其它数据空间大小为5930(M)。<br>2. 建议检查其它空间中各文件的大小。尽量避免超大事务。 |

# 5 SQL分析

## 5.1 慢SQL分析

### 5.1.1 SQL-1

| | |
|---|---|
| SQL语句 | SELECT task_id, garage_id, garage_code, company_code, org_id , task_type, car_type, user_account, user_phone, user_org_code , user_type, survey_number, survey_name, survey_mobile, survey_org , regist_no, regist_date, loss_id, license_no, vehicle_model , vehicle_brand, brand_code, car_owner, mobile, policy_no , address, province_name, city_name, district_name, detail_address , longitude, latitude, remark, batch_status, status , start_time, expected_time, confirm_time, inform_time, finish_time , task_source, ev_time, ev_id, fail_count, fx_garage_id , giveup_reason_code, giveup_reason_name, province_code, city_code, town_code , note, end_msg, repair_type, repair_name, repair_org , car_application, create_by, create_time, update_by, update_time FROM et_repair_task WHERE status = ? AND update_time <= ? AND task_type = ? AND garage_code IS NULL AND regist_no IS NOT NULL AND batch_status = ? AND fail_count < ? ORDER BY update_time LIMIT ? |
| 建议 | 1. **行扫描**<br>行扫描总数:495949568，执行次数:456。<br>2. **order by**<br>该语句使用了"order by"。注意检查是否有效利用到了索引。<br>3. **索引**<br>该语句没有使用聚合函数和模糊查询。行扫描很高且没有记录返回。创建合理索引通常会带来很大的性能提升。 |

### 5.1.2 SQL-2

| | |
|---|---|
| SQL语句 | SELECT loss_item_id, case_info_id, case_no, loss_id, loss_name , loss_type, loss_status, lossitem_status, surveyor_code, surveyor_name , mobile, contact, contact_phone, end_case_time, finish_case_time , remark, user_type, check_loss_usercode, check_loss_username, org_code , org_name, loss_deal_type, vip_level, create_by, create_time , modify_by, modify_time, fill_status FROM prpm_loss_item WHERE mobile = ? AND user_type = ? ORDER BY create_time DESC LIMIT ? |
| 建议 | 1. **行扫描**<br>行扫描与行返回之比为68150125。<br>2. **order by**<br>该语句使用了"order by"。注意检查是否有效利用到了索引。<br>3. **索引**<br>该语句没有使用聚合函数和模糊查询，行扫描与行返回之比很高。创建合理索引会带来很大的优化空间。 |

### 5.1.3 SQL-3

| | |
|---|---|
| SQL语句 | SELECT DISTINCT ui.case_no FROM prpm_upload_image ui JOIN prpm_loss_item l ON ui.case_no = l.case_no WHERE ui.stay_upload NOT IN (?) AND l.loss_status = ? |
| 建议 | 1. **返回记录数**<br>语句最大返回记录数为21。<br>2. **行扫描**<br>行扫描与行返回之比为148842。<br>3. **索引**<br>该语句没有使用聚合函数和模糊查询，行扫描与行返回之比很高。创建合理索引会带来很大的优化空间。 |

### 5.1.4 SQL-4

| | |
|---|---|
| SQL语句 | SELECT DISTINCT ui.* FROM prpm_upload_image ui JOIN prpm_loss_item l ON ui.case_no = l.case_no WHERE ui.stay_upload != ? AND l.loss_status = ? ORDER BY ui.create_time DESC LIMIT ? |
| 建议 | 1. **返回记录数**<br>语句最大返回记录数为50。<br>2. **行扫描**<br>行扫描与行返回之比为49395。<br>3. **order by**<br>该语句使用了"order by"。注意检查是否有效利用到了索引。<br>4. **索引**<br>该语句没有使用聚合函数和模糊查询，行扫描与行返回之比很高。创建合理索引会带来很大的优化空间。 |

### 5.1.5 SQL-5

| | |
|---|---|
| SQL语句 | SELECT l.case_info_cyp_id, l.case_no, ui.* FROM prpm_upload_loss_item_img ui JOIN prpm_loss_item_cyp l ON ui.loss_item_cyp_id = l.loss_item_cyp_id WHERE ui.stay_upload != ? AND (l.handle_status IN (?) OR ui.loss_item_cyp_id IS NULL OR ui.loss_item_cyp_id = ?) ORDER BY ui.create_time DESC LIMIT ? |
| 建议 | 1. **行扫描**<br>行扫描与行返回之比为2629178。<br>2. **order by**<br>该语句使用了"order by"。注意检查是否有效利用到了索引。<br>3. **索引**<br>该语句没有使用聚合函数和模糊查询，行扫描与行返回之比很高。创建合理索引会带来很大的优化空间。 |

### 5.1.6 SQL-6

| | |
|---|---|
| SQL语句 | SELECT DISTINCT l.case_no FROM prpm_upload_loss_item_img ui JOIN prpm_loss_item_cyp l ON ui.loss_item_cyp_id = l.loss_item_cyp_id WHERE ui.stay_upload != ? AND (l.handle_status IN (?) OR ui.loss_item_cyp_id IS NULL OR ui.loss_item_cyp_id = ?) |
| 建议 | 1. **行扫描**<br>行扫描与行返回之比为3533425。<br>2. **索引**<br>该语句没有使用聚合函数和模糊查询，行扫描与行返回之比很高。创建合理索引会带来很大的优化空间。 |

### 5.1.7 SQL-7

| | |
|---|---|
| SQL语句 | SELECT task_id AS taskId, type.type_name AS taskStatus, finish_time AS finishTime, eg.garage_name_abb AS garageName FROM et_repair_task t LEFT JOIN et_garage eg ON t.garage_id = eg.garage_id AND eg.audit_status = ? LEFT JOIN et_type type ON type.type_code = t.`status` LEFT JOIN et_typegroup tg ON tg.type_group_id = type.type_group_id AND tg.type_group_code = ? WHERE eg.garage_code = ? AND t.status NOT IN (?) ORDER BY t.create_time, t.task_id |
| 建议 | 1. **返回记录数**<br>语句最大返回记录数为405。<br>2. **行扫描**<br>行扫描与行返回之比为18256。<br>3. **order by**<br>该语句使用了"order by"。注意检查是否有效利用到了索引。<br>4. **索引**<br>该语句没有使用聚合函数和模糊查询，行扫描与行返回之比很高。创建合理索引会带来很大的优化空间。 |

### 5.1.8 SQL-8

| | |
|---|---|
| SQL语句 | SELECT l.loss_item_id, l.case_info_id, l.case_no, l.loss_id, l.loss_name , l.loss_type, l.loss_status, l.lossitem_status, l.surveyor_code, l.surveyor_name , l.mobile, l.contact, l.contact_phone, l.end_case_time, l.finish_case_time , l.remark, l.user_type, l.check_loss_usercode, l.check_loss_username, l.org_code , l.org_name, l.loss_deal_type, l.create_by, l.create_time, l.modify_by , l.modify_time, l.fill_status, l.click_time, l.vip_level FROM prpm_loss_item l WHERE 1 = 1 AND l.case_no LIKE CONCAT(CONCAT(?, ?), ?) AND l.org_code = ? AND l.create_time <= ? ORDER BY l.create_time DESC LIMIT ?, ? |
| 建议 | 1. **行扫描**<br>行扫描与行返回之比为1303740。<br>2. **order by**<br>该语句使用了"order by"。注意检查是否有效利用到了索引。<br>3. **索引**<br>该语句没有使用聚合函数和模糊查询，行扫描与行返回之比很高。创建合理索引会带来很大的优化空间。 |

### 5.1.9 SQL-9

| | |
|---|---|
| SQL语句 | SELECT l.loss_item_id, l.case_info_id, l.case_no, l.loss_id, l.loss_name , l.loss_type, l.loss_status, l.lossitem_status, l.surveyor_code, l.surveyor_name , l.mobile, l.contact, l.contact_phone, l.end_case_time, l.finish_case_time , l.remark, l.user_type, l.check_loss_usercode, l.check_loss_username, l.org_code , l.org_name, l.loss_deal_type, l.create_by, l.create_time, l.modify_by , l.modify_time, l.fill_status, l.click_time, l.vip_level FROM prpm_loss_item l WHERE l.org_code = ? AND l.create_time <= ? AND l.user_type = ? AND l.loss_status IN (?) ORDER BY l.create_time DESC LIMIT ?, ? |
| 建议 | 1. **行扫描**<br>行扫描与行返回之比为624535。<br>2. **order by**<br>该语句使用了"order by"。注意检查是否有效利用到了索引。<br>3. **索引**<br>该语句没有使用聚合函数和模糊查询，行扫描与行返回之比很高。创建合理索引会带来很大的优化空间。 |

### 5.1.10 SQL-10

| | |
|---|---|
| SQL语句 | SELECT l.loss_item_id, l.case_info_id, l.case_no, l.loss_id, l.loss_name , l.loss_type, l.loss_status, l.lossitem_status, l.surveyor_code, l.surveyor_name , l.mobile, l.contact, l.contact_phone, l.end_case_time, l.finish_case_time , l.remark, l.user_type, l.check_loss_usercode, l.check_loss_username, l.org_code , l.org_name, l.loss_deal_type, l.create_by, l.create_time, l.modify_by , l.modify_time, l.fill_status, l.click_time, l.vip_level FROM prpm_loss_item l WHERE l.org_code = ? AND l.user_type = ? AND l.loss_status IN (?) ORDER BY l.create_time DESC LIMIT ?, ? |
| 建议 | 1. **行扫描**<br>行扫描与行返回之比为624643。<br>2. **order by**<br>该语句使用了"order by"。注意检查是否有效利用到了索引。<br>3. **索引**<br>该语句没有使用聚合函数和模糊查询，行扫描与行返回之比很高。创建合理索引会带来很大的优化空间。 |

## 5.1.11 SQL-11

| | |
|---|---|
| SQL语句 | SELECT l.loss_item_id, l.case_info_id, l.case_no, l.loss_id, l.loss_name , l.loss_type, l.loss_status, l.lossitem_status, l.surveyor_code, l.surveyor_name , l.mobile, l.contact, l.contact_phone, l.end_case_time, l.finish_case_time , l.remark, l.user_type, l.check_loss_usercode, l.check_loss_username, l.org_code , l.org_name, l.loss_deal_type, l.create_by, l.create_time, l.modify_by , l.modify_time, l.fill_status, l.click_time, l.vip_level, info.damage_date , info.end_case_time, info.quick_deal_area, info.report_type FROM prpm_loss_item l JOIN prpm_case_info info ON l.case_no = info.case_no WHERE 1 = 1 AND l.case_no LIKE CONCAT(CONCAT(?, ?), ?) AND info.org_code = ? AND info.damage_date > ? AND info.damage_date < ? AND l.user_type = ? ORDER BY l.create_time DESC LIMIT ?, ? |
| 建议 | 1. **行扫描**<br>行扫描与行返回之比为1665012。<br>2. **order by**<br>该语句使用了"order by"。注意检查是否有效利用到了索引。<br>3. **索引**<br>该语句没有使用聚合函数和模糊查询，行扫描与行返回之比很高。创建合理索引会带来很大的优化空间。 |

## 5.1.12 SQL-12

| | |
|---|---|
| SQL语句 | SELECT user_type AS 'userType' FROM prpm_loss_item p WHERE p.org_code = ? AND to_days(p.create_time) = to_days(?) |
| 建议 | 1. **返回记录数**<br>语句最大返回记录数为587。该语句每次返回记录数过高，需要从业务和设计的角度评估是否可以优化。<br>2. **行扫描**<br>行扫描与行返回之比为1166。<br>3. **查询条件中使用函数**<br>该语句的查询条件中使用了函数"to_days(p.create_time)"。查询条件中对查询字段使用函数会导致SQL语句不能使用到已有索引，应尽量避免。<br>4. **索引**<br>该语句没有使用聚合函数和模糊查询，行扫描与行返回之比很高。创建合理索引会带来很大的优化空间。 |

## 5.1.13 SQL-13

| | |
|---|---|
| SQL语句 | SELECT COUNT(*) FROM (SELECT imgsur.*, loss.loss_id AS 'lossItem.lossId', loss.loss_name AS 'lossItem.lossName', loss.loss_type AS 'lossItem.lossType', loss.vip_level AS 'lossItem.vipLevel' FROM prpm_image_survey imgsur JOIN prpm_loss_item loss ON imgsur.loss_item_id = loss.loss_item_id WHERE loss.org_code = ? AND loss.loss_status IN (?) AND imgsur.user_type != ? AND imgsur.is_dealing = ? AND imgsur.user_code = ? ORDER BY imgsur.create_time DESC ) tmp_count |
| 建议 | 1. **行扫描**<br>行扫描与行返回之比为484226。<br>2. **order by**<br>该语句使用了"order by"。注意检查是否有效利用到了索引。<br>3. **聚合函数**<br>该语句使用了聚合函数。 |

### 5.1.14 SQL-14

| | |
|---|---|
| SQL语句 | SELECT COUNT(*) FROM prpm_loss_item l JOIN prpm_case_info info ON l.case_no = info.case_no WHERE 1 = 1 AND info.org_code = ? AND info.damage_date > ? AND info.damage_date < ? AND l.user_type = ? |
| 建议 | 1. **行扫描**<br>行扫描与行返回之比为570475。<br>2. **聚合函数**<br>该语句使用了聚合函数。 |

### 5.1.15 SQL-15

| | |
|---|---|
| SQL语句 | SELECT COUNT(*) FROM (SELECT imgsur.*, loss.loss_id AS 'lossItem.lossId', loss.loss_name AS 'lossItem.lossName', loss.loss_type AS 'lossItem.lossType', loss.vip_level AS 'lossItem.vipLevel' FROM prpm_image_survey imgsur JOIN prpm_loss_item loss ON imgsur.loss_item_id = loss.loss_item_id WHERE loss.org_code = ? AND loss.loss_status IN (?) AND imgsur.user_type != ? AND imgsur.is_dealing = ? AND imgsur.user_code = ? AND imgsur.case_no LIKE CONCAT(CONCAT(?, ?), ?) ORDER BY imgsur.create_time DESC ) tmp_count |
| 建议 | 1. **行扫描**<br>行扫描与行返回之比为458126。<br>2. **order by**<br>该语句使用了"order by"。注意检查是否有效利用到了索引。<br>3. **聚合函数**<br>该语句使用了聚合函数。执行次数并不高，但单次执行时间超过5秒。需要注意该大查询对系统整体性能的影响。 |

## 5.1.16 SQL-16

| | |
|---|---|
| SQL语句 | SELECT task_id AS taskId, type.type_name AS taskStatus, finish_time AS finishTime, eg.garage_name_abb AS garageName FROM et_repair_task t LEFT JOIN et_garage eg ON t.garage_id = eg.garage_id AND eg.audit_status = ? LEFT JOIN et_type type ON type.type_code = t.`status` LEFT JOIN et_typegroup tg ON tg.type_group_id = type.type_group_id AND tg.type_group_code = ? WHERE eg.garage_code = ? AND t.task_id LIKE concat(?, ?) AND t.status NOT IN (?) ORDER BY t.create_time, t.task_id |
| 建议 | 1. **行扫描**<br>行扫描总数:3461185，执行次数:3。<br>2. **order by**<br>该语句使用了"order by"。注意检查是否有效利用到了索引。<br>3. **索引**<br>该语句没有使用聚合函数和模糊查询。行扫描很高且没有记录返回。创建合理索引通常会带来很大的性能提升。 |

## 5.1.17 SQL-17

| | |
|---|---|
| SQL语句 | SELECT l.loss_item_id, l.case_info_id, l.case_no, l.loss_id, l.loss_name , l.loss_type, l.loss_status, l.lossitem_status, l.surveyor_code, l.surveyor_name , l.mobile, l.contact, l.contact_phone, l.end_case_time, l.finish_case_time , l.remark, l.user_type, l.check_loss_usercode, l.check_loss_username, l.org_code , l.org_name, l.loss_deal_type, l.create_by, l.create_time, l.modify_by , l.modify_time, l.fill_status, l.click_time, l.vip_level, info.damage_date , info.end_case_time, info.quick_deal_area, info.report_type FROM prpm_loss_item l JOIN prpm_case_info info ON l.case_no = info.case_no WHERE 1 = 1 AND l.case_no LIKE CONCAT(CONCAT(?, ?), ?) AND info.org_code = ? AND l.user_type = ? ORDER BY l.create_time DESC LIMIT ?, ? |
| 建议 | 1. **行扫描**<br>行扫描与行返回之比为1666246。<br>2. **order by**<br>该语句使用了"order by"。注意检查是否有效利用到了索引。<br>3. **索引**<br>该语句没有使用聚合函数和模糊查询，行扫描与行返回之比很高。创建合理索引会带来很大的优化空间。 |

## 5.1.18 SQL-18

| | |
|---|---|
| SQL语句 | SELECT imgsur.*, loss.loss_id AS 'lossItem.lossId', loss.loss_name AS 'lossItem.lossName', loss.loss_type AS 'lossItem.lossType', loss.vip_level AS 'lossItem.vipLevel' FROM prpm_image_survey imgsur JOIN prpm_loss_item loss ON imgsur.loss_item_id = loss.loss_item_id WHERE loss.org_code = ? AND loss.loss_status IN (?) AND imgsur.user_type != ? AND imgsur.is_dealing = ? AND imgsur.user_code = ? AND imgsur.case_no LIKE CONCAT(CONCAT(?, ?), ?) ORDER BY imgsur.create_time DESC LIMIT ?, ? |
| 建议 | 1. **行扫描**<br>行扫描与行返回之比为296819。<br>2. **order by**<br>该语句使用了"order by"。注意检查是否有效利用到了索引。<br>3. **索引**<br>该语句没有使用聚合函数和模糊查询，行扫描与行返回之比很高。创建合理索引会带来很大的优化空间。 |

## 5.1.19 SQL-19

| | |
|---|---|
| SQL语句 | SELECT a.case_no AS caseNo, a.license_no AS licenseNo, loss.loss_status AS caseStatus, a.create_time AS createTime, a.end_case_time AS endCaseTime , a.org_id AS orgId, loss.surveyor_code AS surveyorCode, loss.surveyor_name AS surveyorName, loss.loss_deal_type AS lossDealType, a.billing_status AS billingStatus , loss.loss_name AS lossName, loss.loss_id AS lossId, loss.loss_type AS lossType, loss.org_code AS orgCode, loss.org_name AS orgName , loss.mobile AS mobile, loss.check_loss_usercode AS checkLossUsercode, loss.check_loss_username AS checkLossUsername, loss.user_type AS userType FROM prpm_case_info a RIGHT JOIN prpm_loss_item loss ON loss.case_no = a.case_no WHERE 1 = 1 AND a.case_no LIKE CONCAT(CONCAT(?, ?), ?) AND a.create_time < ? ORDER BY a.create_time DESC LIMIT ?, ? |
| 建议 | 1. **行扫描**<br>行扫描与行返回之比为1426218。<br>2. **order by**<br>该语句使用了"order by"。注意检查是否有效利用到了索引。<br>3. **索引**<br>该语句没有使用聚合函数和模糊查询，行扫描与行返回之比很高。创建合理索引会带来很大的优化空间。 |

## 5.1.20 SQL-20

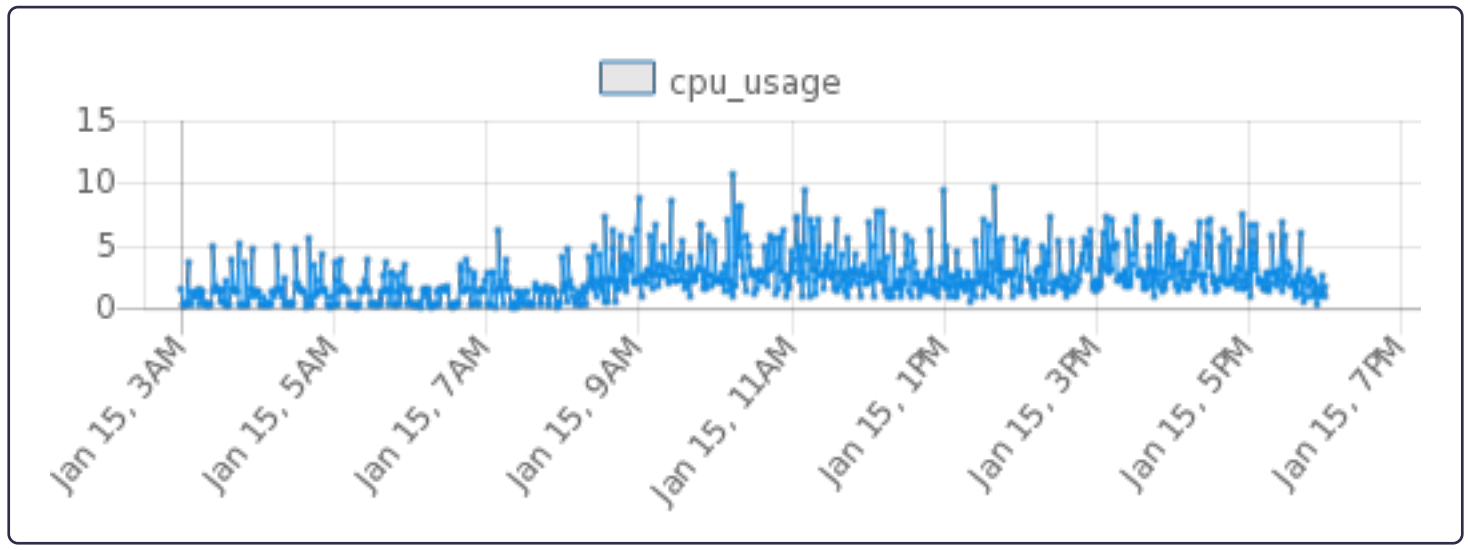
| | |
|---|---|
| SQL语句 | SELECT COUNT(*) FROM prpm_case_info a RIGHT JOIN prpm_loss_item loss ON loss.case_no = a.case_no WHERE 1 = 1 AND a.case_no LIKE CONCAT(CONCAT(?, ?), ?) AND a.create_time < ? |
| 建议 | 1. **行扫描**<br>行扫描与行返回之比为1426216。<br>2. **聚合函数**<br>该语句使用了聚合函数。执行次数并不高，但单次执行时间超过5秒。需要注意该大查询对系统整体性能的影响。 |

## 5.1.21 SQL-21

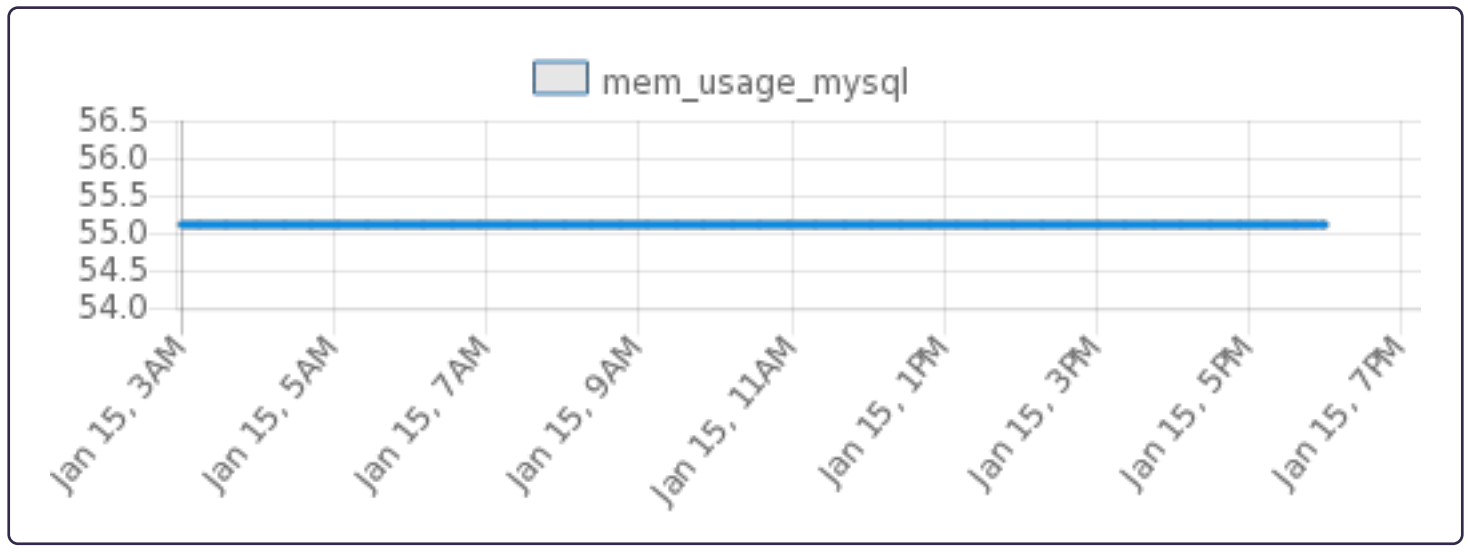
| | |
|---|---|
| SQL语句 | SELECT b.billing_info_id, b.case_no, b.billing_type, b.billing_status, b.org_code , b.create_time, b.create_by, b.modify_time, b.modify_by, o.org_name FROM prpm_billing_info b LEFT JOIN prpm_org o ON o.org_code = b.org_code WHERE 1 = 1 AND b.case_no LIKE concat(?, ?, ?) AND b.create_time >= ? AND b.create_time <= ? ORDER BY b.create_time DESC LIMIT ?, ? |
| 建议 | 1. **行扫描**<br>行扫描与行返回之比为1359120。<br>2. **order by**<br>该语句使用了"order by"。注意检查是否有效利用到了索引。<br>3. **索引**<br>该语句没有使用聚合函数和模糊查询，行扫描与行返回之比很高。创建合理索引会带来很大的优化空间。 |

# 6 性能曲线

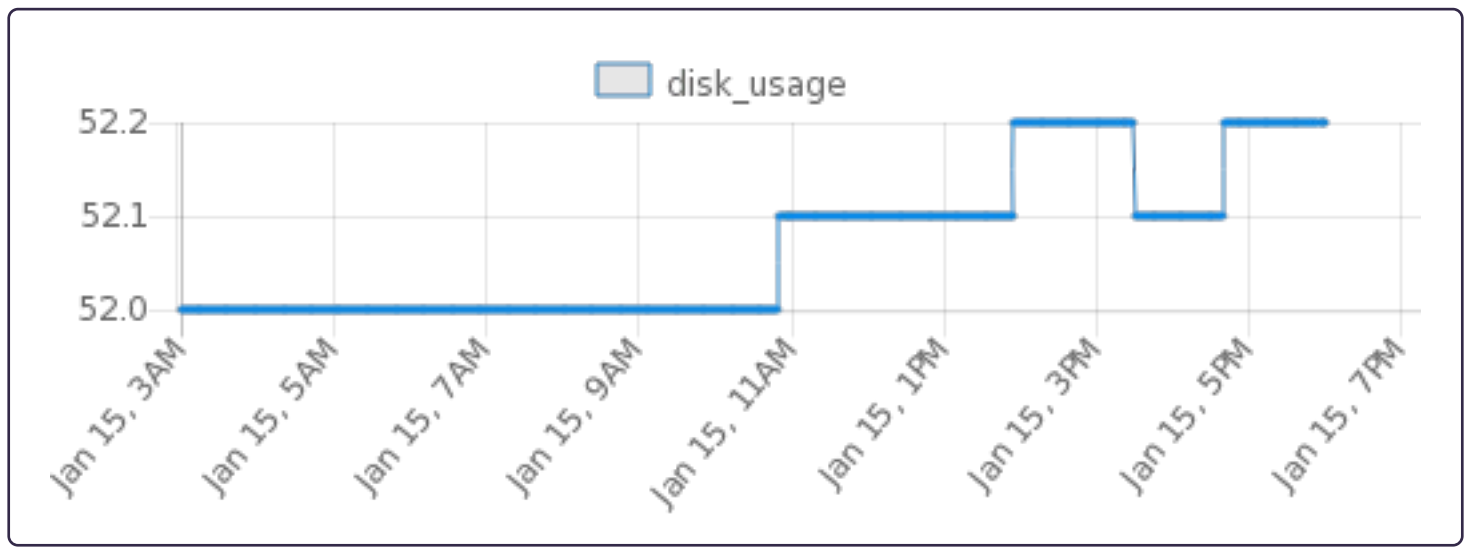
## 6.1 CPU利用率



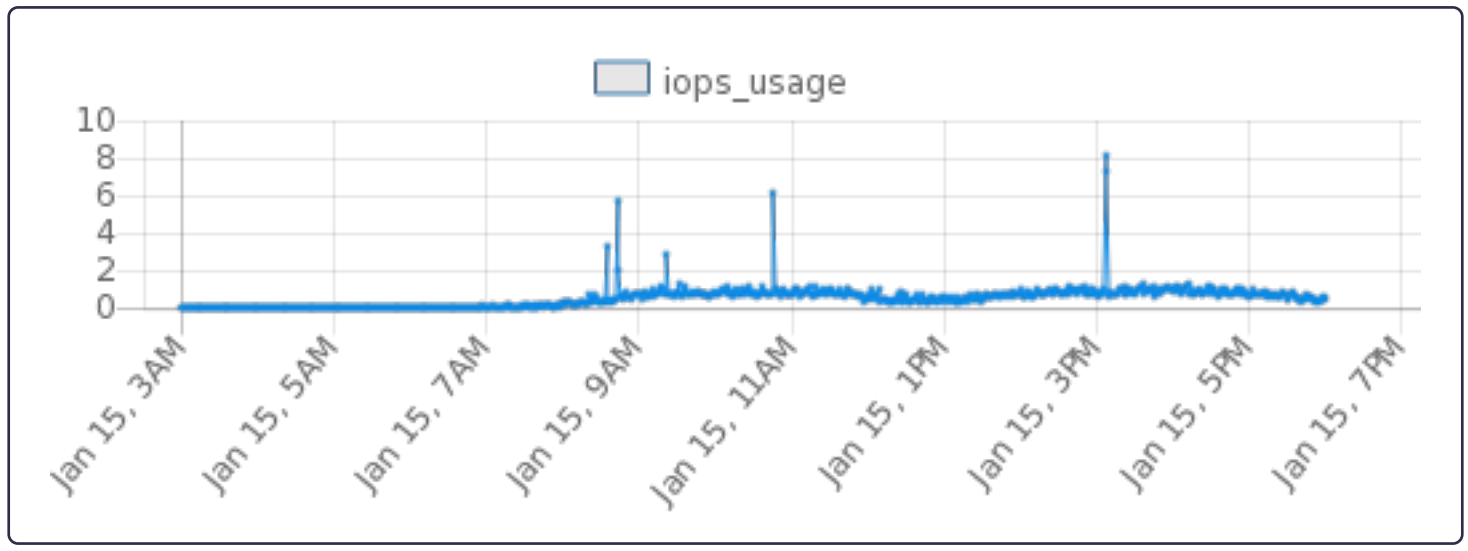

## 6.2 内存利用率



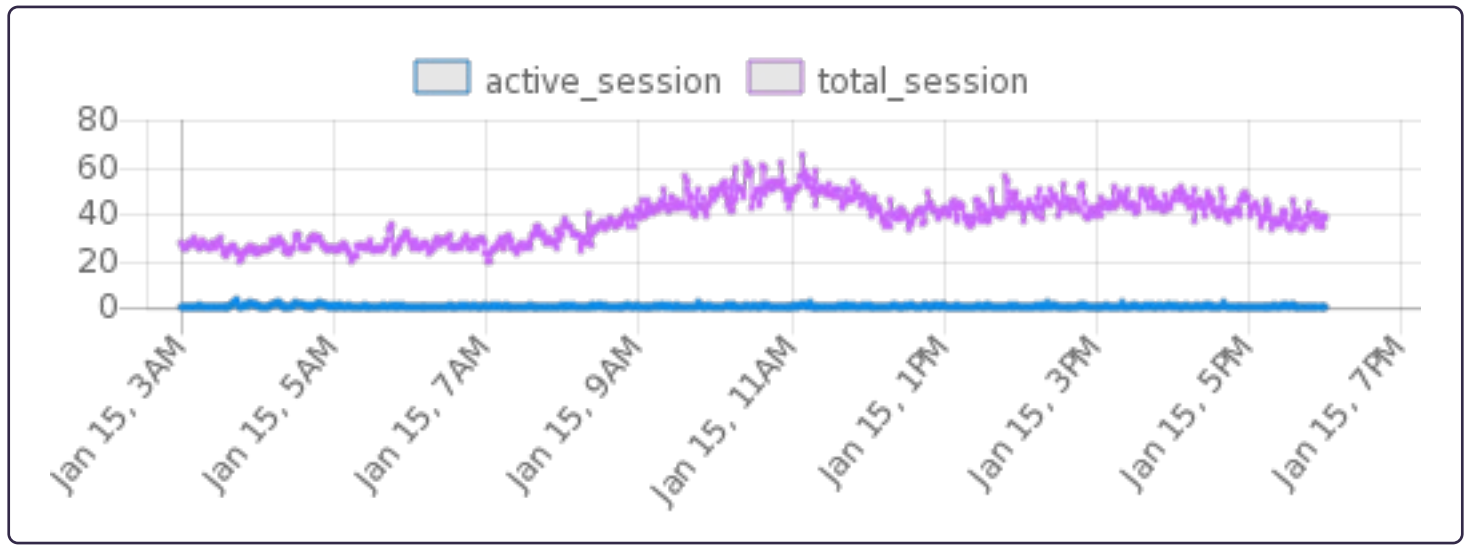

## 6.3 磁盘空间



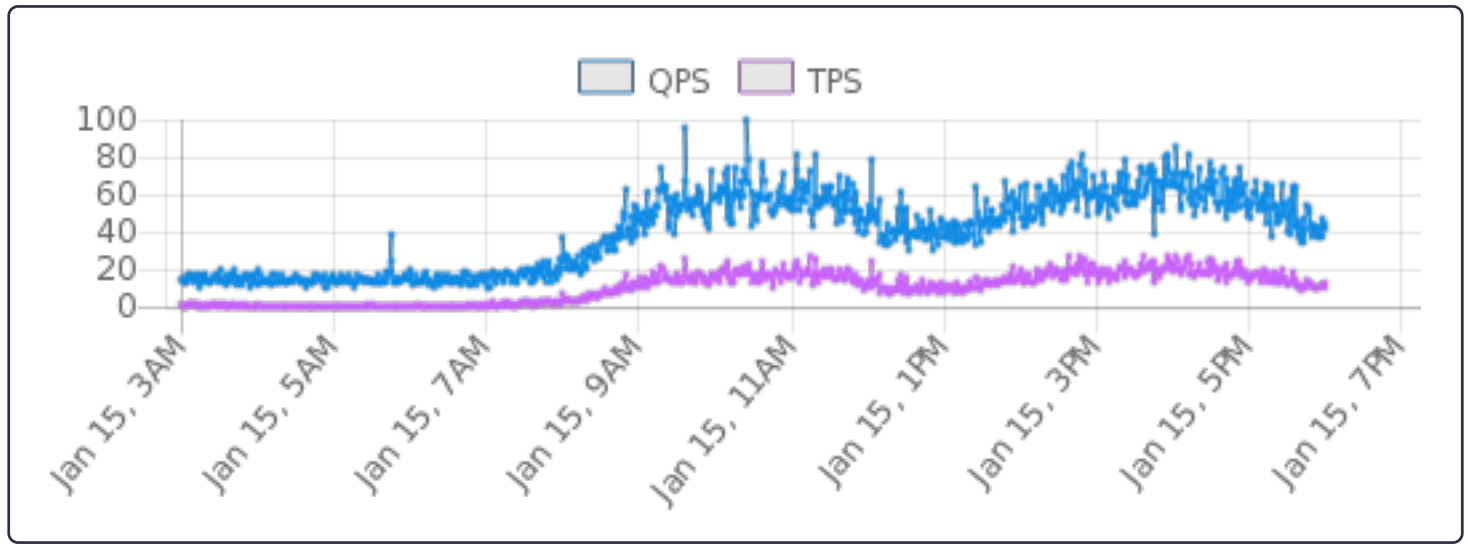
## 6.4 IOPS利用率




## 6.5 会话




## 6.6 QPSTPS




## 6.7 InnoDB